PERGAMON

Computers & Graphics 25 (2001) 67–75

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Shape Blending

Guaranteed intersection-free polygon morphing

Craig Gotsman*, Vitaly Surazhsky

Faculty of Computer Science, Technion–Israel Institute of Technology, Haifa 32000, Israel

Abstract

We present a method for naturally and continuously morphing two simple planar polygons with corresponding vertices in a manner that *guarantees* that the intermediate polygons are also simple. This contrasts with all existing polygon morphing schemes who cannot guarantee the non-self-intersection property on a global scale, due to the heuristics they employ. Our method achieves this property by reducing the polygon morphing problem to the problem of morphing compatible planar triangulations of corresponding point sets, which is performed by interpolating vertex barycentric coordinates instead of vertex locations. The reduction involves compatibly triangulating simple polygons and polygons with a single hole. We show how to achieve this using only a small number of extra (Steiner) vertices. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Morphing; Metamorphosis; Blending; Polygons; Geometric modeling

1. Introduction

Morphing, also known as metamorphosis, is the gradual transformation of one shape (the *source*) into another (the *target*). Morphing has wide practical use in areas such as computer graphics, animation and modeling. To achieve more spectacular, impressive and accurate results, the morphing process requires a lot of the work to be done manually. A major research challenge is to develop techniques that will automate this process as much as possible.

The morphing problem has been investigated in many contexts, e.g., morphing of two-dimensional images [1–3], polygons and polylines [4–9], freeform curves [10] and even voxel-based volumetric representations [11]. The morphing process always consists of solving two main problems. The first one is to find a correspondence between elements (features) of the two shapes. The second problem is to find trajectories that corresponding elements traverse during the morphing process. Regrettably, a formal definition of a successful correspondence does not exist, as well as a definition of a successful solution of the trajectory problem. In this paper we assume the more typical morphing scenario, namely, that the correspondence is given and only the trajectory problem is to be solved.

The naive approach to solve the trajectory problem is to choose the trajectories to be straight lines, where every feature of the shape travels with a constant velocity along its line towards the corresponding feature of the target during the morph. Unfortunately, this simple approach can lead to undesirable results. The intermediate shapes can vanish, i.e. degenerate into a single point, or contain self-intersections even though the source and target are simple, namely, self-intersection-free. Even if the linear morph is free of self-intersections and degeneracies, its intermediate shapes may have areas or distances between features far from those of the source and target, resulting in a "misbehaved" looking morph. Most of the research on solving the trajectory problem for morphing concentrates on trying to eliminate self-intersections and preserve the geometrical properties of the intermediate shapes. Many existing methods achieve good results for many inputs, however, none guarantees that the resulting morph is always self-intersection-free. One exception is the method of Guibas et al. [12], but this works only for

* Corresponding author. Tel.: + 972-4-829-4336; fax: + 972-4-822-1128.

E-mail addresses: gotsman@cs.technion.ac.il (C. Gotsman), vitus@cs.technion.ac.il (V. Surazhsky).

0097-8493/01/$ - see front matter © 2001 Elsevier Science Ltd. All rights reserved.
PII: S0097-8493(00)00108-4

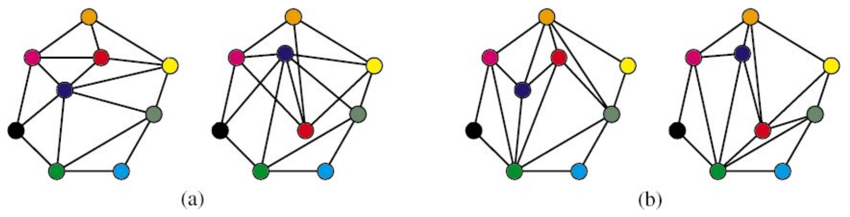

Fig. 1. (a) Corresponding point sets (correspondence denoted by colored vertices) with incompatible triangulation. (b) Same point sets with compatible triangulation.

so-called "parallel" source and target polygons, i.e. those whose corresponding edges are parallel.

In this paper we present a polygon morphing scheme, which, to the best of our knowledge, is the only one which guarantees that the intermediate polygons are simple, given any simple source and target polygons. This is achieved by reducing the problem to planar triangulation morphing, for which morphing techniques which preserve triangle orientations have been developed. To perform the reduction, it is necessary to compatibly triangulate the interiors of the source and target polygons, and certain regions exterior to the polygons. We show how to do this efficiently.

Our work is closest in spirit to that of Tal and Elber [3] and Alexa et al. [4], in that they also triangulate the interior of the source and target polygons in a compatible manner prior to morphing. The main motivation for this is that texture mapping be well defined on the interiors of the triangulated polygons as a piecewise-affine mapping. However, the actual vertex trajectories used by Tal and Elber are the simple linear ones, hence do not guarantee that the triangulated polygons are compatible and certainly do not guarantee that they do not self-intersect throughout the morph. Alexa et al. compatibly triangulate the source and target polygons, and optimize the vertex trajectories during the morph in an attempt to maintain compatibility and preserve shape. This results in non-linear vertex trajectories, in which individual triangle orientations seem to be preserved on a local scale throughout the morph. However, on a global scale, there is nothing to prevent the intermediate polygons from self-intersecting.

The paper is organized as follows: Section 2 introduces the notion of compatible triangulations and reviews previous relevant work on morphing of compatible triangulations. Section 3 shows how the polygon morphing problem may be reduced to compatible triangulation morphing by introducing Steiner vertices. Section 4 presents a novel technique to compatibly triangulate the interiors of two simple polygons and a pair of polygons with a single hole each using a small number of Steiner vertices, which is of independent interest. Section 5 shows some results generated by our method, and we conclude in Section 6.

2. Morphing planar triangulations

Given two corresponding point sets, in many applications it is desirable to impose a common graph connectivity on the two point sets. In the case of a triangulated graph, this means that the common connectivity should result in planar straight-line triangulations of both point sets and we then say that the common connectivity is a triangulation *compatible* with both point sets. An immediate application of compatible triangulations (also known as *isomorphic* or *joint triangulations*) is that a bijective piecewise-affine mapping (also known as a *homeomorphic simplicial map*), interpolating the point-to-point correspondence, may be defined on the convex hulls of the two point sets, without "holes" or "overlaps" in the mapping (see Fig. 1). This topic has been investigated in computational geometry, and the decision problem whether two given corresponding point sets admit a compatible triangulation is believed to be NP-complete [13]. In the scenario where a compatible triangulation is already given, Floater and Gotsman [14] first introduced a general method for continuous morphing between two given compatible planar triangulations *with a common convex boundary*. The method is based on the convex representation of triangulations using barycentric coordinates, first introduced by Tutte [15] for graph drawing purposes, and later generalized by Floater [16] for parameterization of 3D meshes. In a nutshell, this means that a planar triangulation on n points may be (non-uniquely) represented by a sparse $n \times n$ stochastic matrix (i.e. a non-negative matrix with unit row sums), where the ith row contains non-zero entries only at columns associated with neighbors of the ith vertex. The geometry (locations) of the point set is the eigenvector of the matrix with unit eigenvalue. At the heart of the morphing method lies the principle that the morphing interpolation procedure should not be done in shape space, rather in matrix space. This avoids many of the problems associated with morphing, and guarantees that the triangulation remains valid (i.e. compatible with the source and target) throughout the morph (see Fig. 2). The simplest morph in matrix space is linear interpolation of the corresponding matrices, since, obviously, the interpolated matrices will also be stochastic, hence represent triangulations. So even

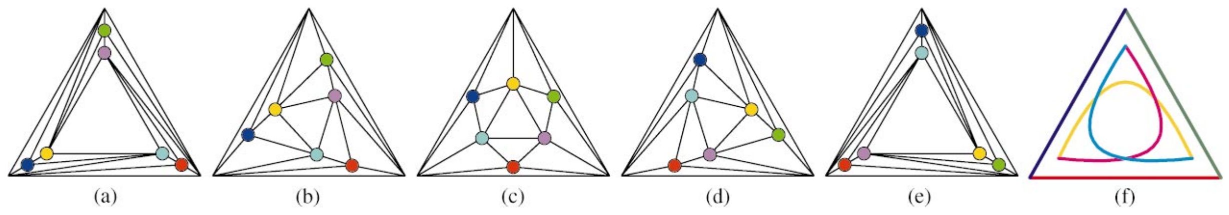

Fig. 2. Morphing compatible triangulations with fixed convex boundary: (a) Source triangulation. (e) Target triangulation. Colored vertices denote correspondence. (b)–(d) Intermediate triangulations generated using the methods of [9]. (f) Morph vertex trajectories.

though the morph is quite trivial in matrix space, it can be quite complex in triangulation space, and yield quite interesting results. The basic method, however, is global in nature, and does not permit local control of the morph. This has been recently addressed by Surazhsky and Gotsman [17][1], and more local control of the individual vertex trajectories is now possible.

These matrix space techniques always yield a continuous morph and are a robust method for morphing images represented by textured triangulations. Regrettably, they are appropriate only for planar triangulations with identical convex boundaries, hence not immediately applicable to polygon morphing.

3. Morphing planar polygons

Since triangulation morphing requires a common convex boundary in the source and target, that method cannot be used in a straightforward manner to morph polygons. The correct way to morph polygons would be to embed them in corresponding triangulations as a cycle of interior edges.

A contiguous sequence of edges within a triangulation, namely, a path in the graph of triangulation, may be viewed as a piecewise linear curve. A simple closed path forms a polygon. Hence, a morph between two corresponding polylines or polygons embedded into two compatible triangulations may be generated by morphing between these triangulations. Clearly, any valid morph between these two triangulations will define a valid morph of the embedded polylines. However, this requires us to find embeddings that allow this.

So, given two n-gons with a correspondence between their vertices, we are required to find two compatible triangulations with identical boundaries such that the two given polygons are embedded into the corresponding triangulations and these embeddings correspond. Only then is the triangulation morphing applied. Note that if the cardinality of the two vertex sets is not equal, the smaller vertex set may be complemented by introducing dummy vertices along the polygon edges. The resulting partial correspondence between the source and target vertices may be then completed by the user, or done automatically using dynamic programming techniques [18] optimizing some shape similarity measure.

To morph two planar polygons P_1 and P_2, the general procedure is as follows (see Fig. 3):

(1) Compatibly triangulate the interiors of P_1 and P_2.
(2) Enclose P_1 and P_2 by convex polygons Q_1 and Q_2, respectively, such that the vertices of Q_1 and Q_2 correspond.
(3) Compatibly triangulate the annuli between Q_1 and P_1 and between Q_2 and P_2, respectively.
(4) Morph the two compatible triangulations using the methods of [14] and/or [17].
(5) Discard all edges of the two triangulations not belonging to P_1 and P_2.

Steps 1 and 3 are easier said than done. It is not always possible to compatibly triangulate two polygons or annuli without introducing extra vertices, also known as *Steiner* vertices. Aronov et al. [19] have proposed two algorithms that construct compatible corresponding triangulations of simple n-gons using at most $O(n^2)$ Steiner vertices. In a subsequent work, Babikov et al. [20] construct compatible triangulations of polygons with holes. However, both these methods may generate unfair triangulations as a result of producing skinny (close to degenerate) triangles. They also do not attempt to minimize the number of Steiner vertices, which is desirable. Too many Steiner vertices increase the complexity of the problem by generating larger sets of equations, and impose more constraints on the system, since more vertex trajectories must not intersect. See Fig. 3 for an example of steps 1–3 of our algorithm.

4. Compatible triangulations of polygons

As we have seen in the previous sections, an important ingredient in morphing simple planar polygons is being

[1] Also available at http://www.cs.technion.ac.il/~gotsman/pubs/surazhsky.zip and as V. Surazhsky's Master's thesis, Computer Science Department, Technion–Israel Institute of Technology.

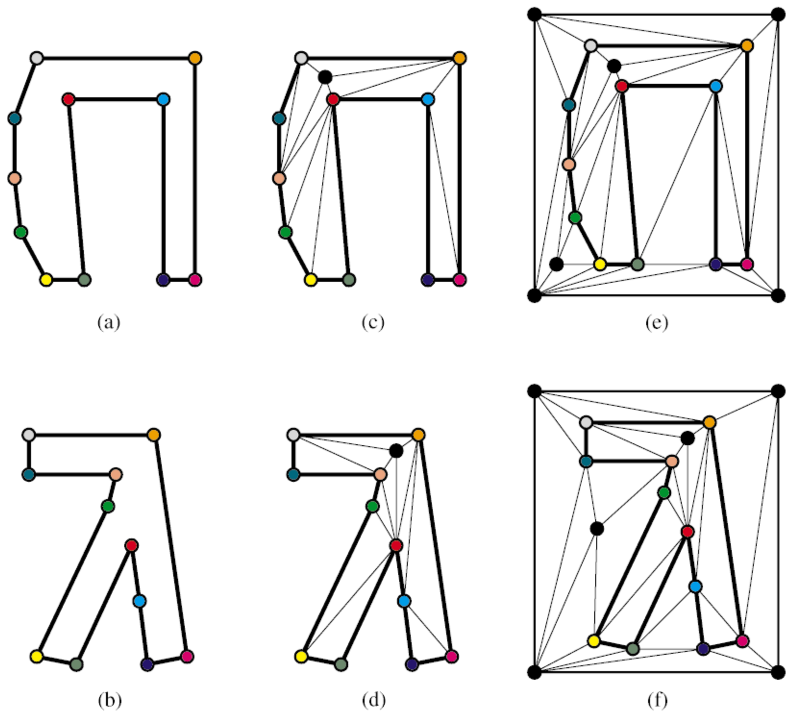

Fig. 3. Embedding polygons in compatible triangulations. (a) Source polygon. (b) Target polygon. (c), (d) Compatible triangulations of (a) and (b). Note that one Steiner (black) vertex is used in the polygon's interiors in order to achieve compatibility. (e), (f) Each polygon is enclosed by a rectangle and the annuli compatibly triangulated. Here too one Steiner vertex is used.

able to compatibly triangulate the source and target polygon interiors and exteriors, preferable with a small number of Steiner vertices. In this section we present a novel algorithm to achieve this goal, having the following characteristics:

- The number and location of Steiner vertices correlates with the relative shape, size and similarity of the input polygons.
- The compatible triangulations do not contain degenerate small and skinny triangles.
- The number of Steiner vertices is small, but not too small, since the optimum may result in degenerate triangles.

In the next two subsections we show how to compatibly triangulate the interiors of two simple polygons, and how to compatibly triangulate the annuli between the given polygons and certain convex enclosures. These annuli are polygons with a single hole, i.e. genus 1.

4.1. Simple polygons

The method we present is an improvement of the second method for building compatible triangulations of polygons described by Aronov et al. [19], based on a so-called *universal* triangulation. This universal triangulation has the form of a *spiderweb* consisting of concentric layers of n-gons and one central vertex, where all corresponding vertices on adjacent layers are connected by radial edges. Our variant, however, may have incomplete concentric layers and, thus, different radial path lengths from the polygon vertices to the central vertex. This *reduced* spiderweb will reduce the number of Steiner vertices by exploiting the shape and similarity of the polygons. This contrasts with the method of Kranakis and Urrutia [21] for compatibly triangulating polygons with a small number of Steiner vertices, exploiting the reflex vertices of the polygons, which might result in degenerate triangles and does not account for the shape similarity between the polygons.

Given a cyclic sequence of n non-negative integers, the *spiderweb connectivity* associated with this sequence is a graph of incomplete concentric cycles such that the *depth* (distance from the central vertex) of the boundary vertices is the given sequence (see Fig. 4(a)) for an example.

Our compatible triangulation algorithm proceeds as follows: Given a simple polygon, we may triangulate it and consider the dual graph of the triangulation, which

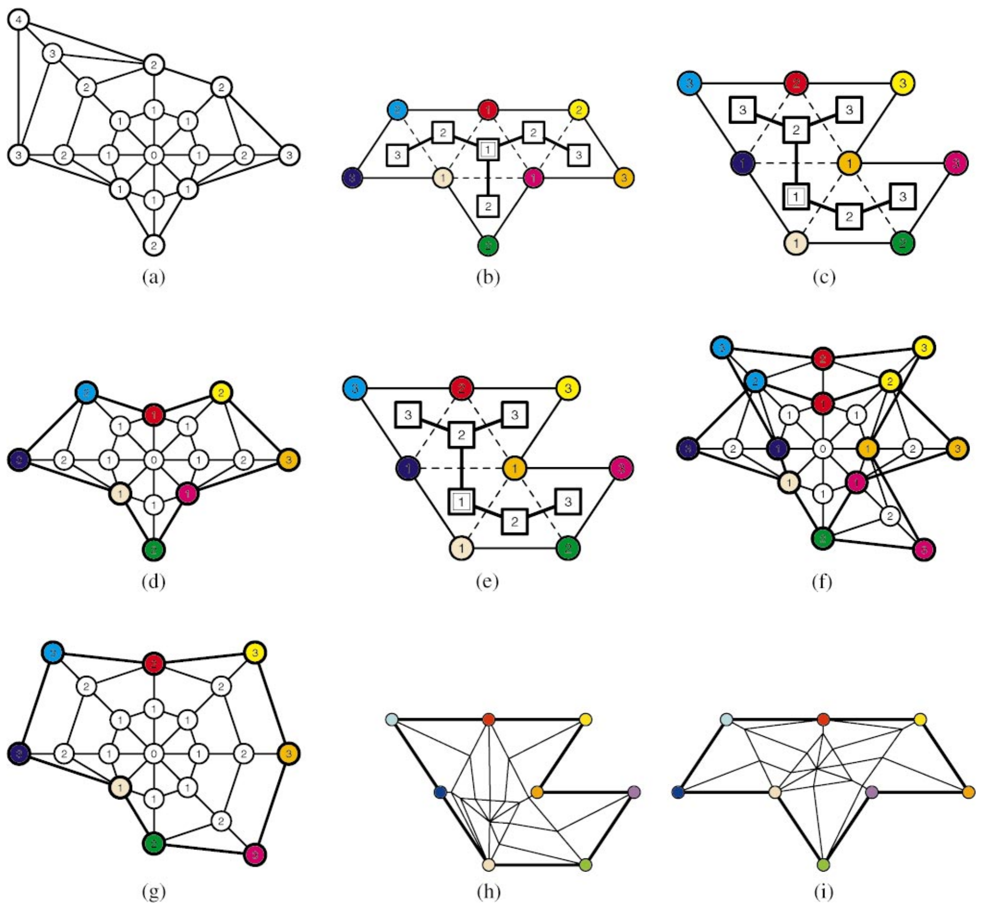

Fig. 4. Construction of compatible triangulations for two simple polygons using a small number of Steiner points. (a) Spiderweb connectivity of cyclic sequence ⟨1,2,2,3,2,2,4,3⟩. (b), (c) Input polygon triangulations and dual trees with face depths and vertex milestones. Depth of central vertex is framed. Given correspondence between polygon vertices is color coded. (d), (e) Spiderweb connectivities corresponding to milestone sequences of (b), (c), respectively. (f) Merged spiderweb connectivity with maximal depths. (g) Common spiderweb connectivity for depth sequence of (f). (h), (i) Compatible triangulation of input polygons based on spiderweb connectivity of (g). Note that the quadrilateral faces may be easily triangulated in a compatible manner due to their convexity.

is always a tree. For the dual graph, we locate a *central* vertex by cutting all tree leaves (equivalent to *ear-cutting* in the triangulation) simultaneously until one vertex — the center — remains. The depth of each triangulation face (dual vertex) is then computed, and subsequently *milestones* for the polygon vertices, defined to be the minimal depth associated with incident triangulation faces (see Fig. 4(b) and (c)).

We then use the cyclic milestone sequence of each of the two input polygons as inputs for constructing the spiderweb connectivities associated with the polygons (see Fig. 4(d) and (e)). The two spiderweb connectivities are merged and a new sequence of (maximal) depths for corresponding vertices is computed (see Fig. 4(f)).

A new (common) spiderweb connectivity based on the new sequence of depths is constructed (see Fig. 4(g)), and then plane graphs with boundaries equivalent to the input polygons and interior Steiner vertices having the same connectivity as the new spiderweb connectivity (see Fig. 4(h) and (i)).

Since both constructed triangulations have the same connectivity due to the common depths, the triangulations are compatible. Note that the algorithm reduces the number of Steiner vertices relative to a regular spiderweb by choosing the depths to be the minimal allowed for both polygons.

4.2. Polygons with a single hole

Babikov et al. [20] present an algorithm for constructing compatible triangulations of polygons with holes. The idea behind their algorithm is to reduce this problem to the one discussed in the previous subsection, namely, compatible triangulations of simple polygons. This is

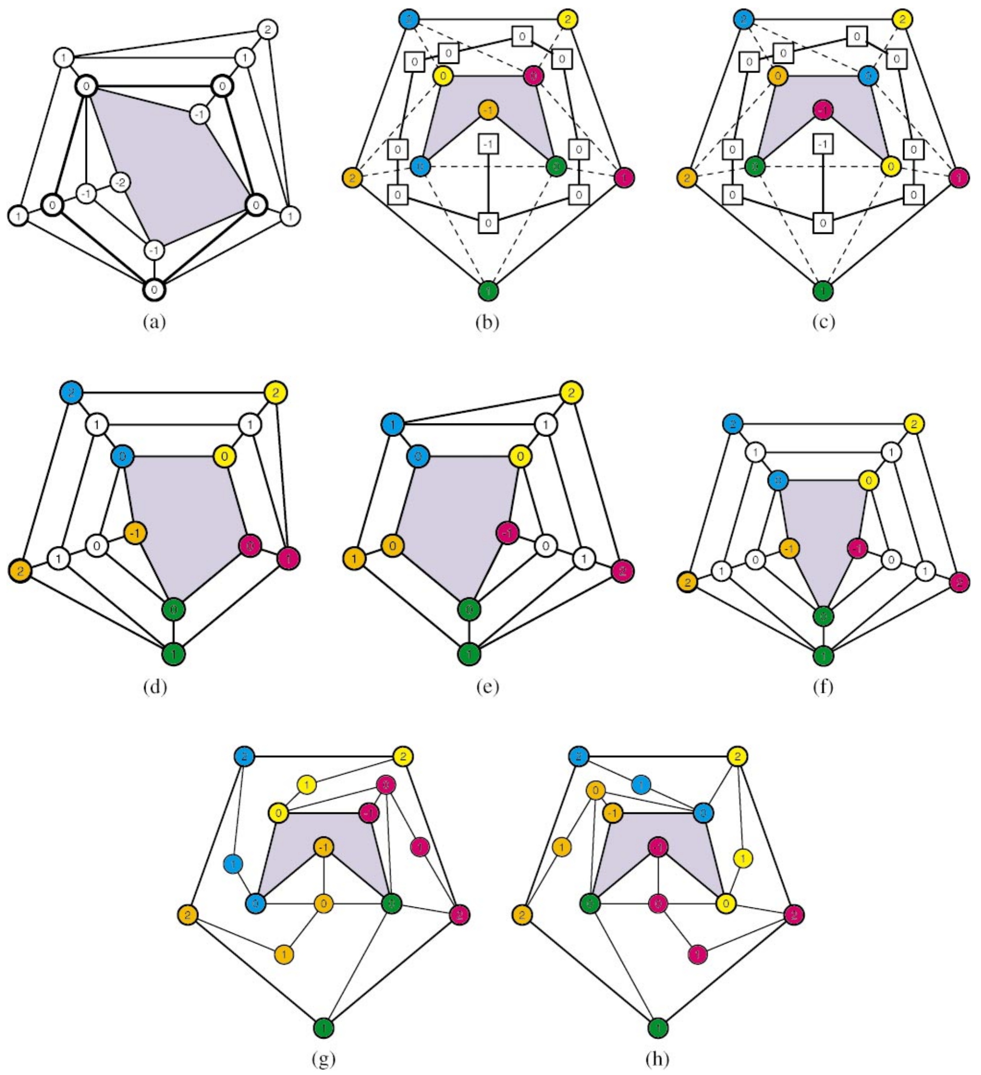

Fig. 5. Construction of compatible triangulations for two simple polygons with single hole using a small number of Steiner points. (a) Spiderweb connectivity of cyclic sequences $\langle 1, 2, 1, 0, 1\rangle$ and $\langle 0, -1, 0, -1, -2\rangle$. The colors denote the correspondence between the two sequences. (b), (c) Annuli triangulations of the input polygons and dual graphs with face depths and vertex milestones. Both polygon boundaries consist of the same number of vertices, with a correspondence between them, denoted by the colors. (d), (e) Spiderweb connectivities corresponding to milestone sequences of (b), (c) respectively. (f) Merged spiderweb connectivity with maximal depths. (g), (h) Compatible tiling of input polygons based on spiderweb connectivity of (f). Note that the non-triangular faces may be triangulated in a compatible manner using our method of Section 4.1.

achieved by cutting the polygons with holes such that simple polygons result. Note that in our case, when the polygon has a single hole, transforming it to a simple polygon by cutting merely opens the single band that was the polygon. Applying techniques for finding compatible triangulations for such an opened band results in a high concentration of Steiner vertices in the region of polygons opposite the cut, which is artificial and wasteful.

We propose a method for finding compatible triangulations of polygons with a single hole which avoids the drawbacks of the simple method by directly applying the spiderweb concept to the polygons without cutting. For this we need to extend the notion of *spiderweb connectivity* to a *pair* of sequences, representing the interior and exterior boundaries of the polygon. Given two cyclic sequences of n integers, the spiderweb connectivity associated with these sequences is a graph of incomplete concentric cycles such that the *depths* (distance from some central loop) of the interior and exterior boundary vertices are the given sequences (see Fig. 5(a) for an example).

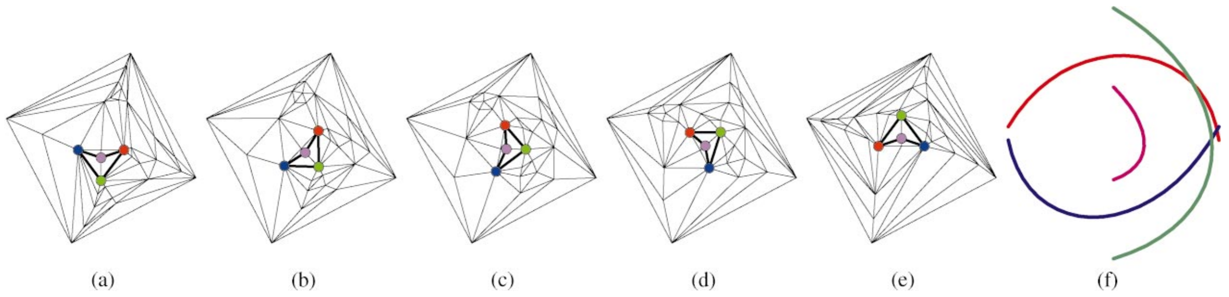
Fig. 6. Morphing simple quadrilaterals: (a) Source polygon embedded in triangulation generated by our algorithm. (e) Embedded target polygon. Colored vertices denote correspondence. Non-colored vertices are (exterior) Steiner vertices. (b), (c), (d) Intermediate (triangulations and) polygons generated using our methods. (f) Morph vertex trajectories.

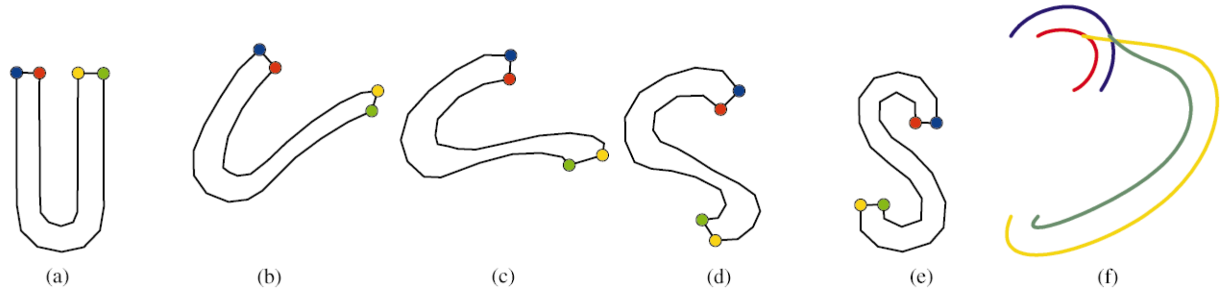
Fig. 7. Morphing simple polygons: (a) Source polygon. (e) Target polygon. Colored vertices denote some of the correspondence. (b), (c), (d) Intermediate polygons generated using our methods. (f) Morph vertex trajectories.

Assume the source and target polygons have the same number of vertices on the interior and exterior boundaries, with some correspondence between these boundaries. We first triangulate both annuli and build the corresponding dual graphs. Note that the dual graphs have exactly one loop due to the connectivity, which we denote by depth 0. Depths of triangles interior to this loop will be negative, and depths of triangles exterior to the loop will be positive. *Milestones* for the interior polygon vertices are defined to be the minimal depth associated with incident triangulation faces. Milestones for exterior vertices are defined as the difference between the topological distance between corresponding interior and exterior vertices in the dual, and the interior vertex milestone (see Fig. 5(b) and (c)).

We then use the cyclic milestone sequences of each of the two input polygon boundaries as inputs for constructing the spiderweb connectivities associated with the polygons (see Fig. 5(d) and (e)). The two spiderweb connectivities are merged and a new (common) spiderweb connectivity based on the new sequence of depths is constructed (see Fig. 5(f)). Plane graphs with boundaries equivalent to the input polygons and interior Steiner vertices having the same connectivity as the new spiderweb connectivity may then be constructed (see Fig. 5(g) and (h)).

Note that the faces of these compatible graphs are not necessarily triangular. If so, these (non-convex) faces may be compatibly triangulated using the methods of Section 4.1.

5. Experimental results

In this section, we demonstrate a number of sample morphs generated by our algorithm. Each morph is illustrated by a number of snapshots during the morph. Animated GIFs of these morphs may be found at http://www.cs.technion.ac.il/gotsman/poly_morphs along with the analogous linear morphs.

The first example is a simple rotation of a non-convex quadrilateral (see Fig. 6). While many methods (including [4]) solve for global transformations between the source and target, e.g. translation, rotation using some sort of optimization procedure, this is unnecessary using our technique. The system automatically “detects” the rotation, without the user having to hint at it, as it is the only possible transformation which preserves both the given

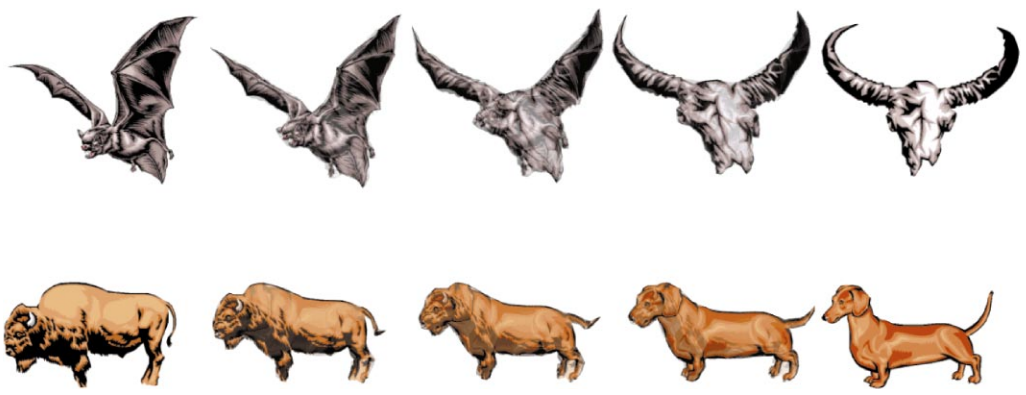
Fig. 8. Morph sequences of texture mapped polygons generated by our algorithm. Top: Input polygons of 33 vertices. The morph uses 35 interior and 359 exterior Steiner vertices. Bottom: Input polygons of 49 vertices. The morph uses 112 interior and 1179 exterior Steiner vertices.

vertex correspondence and compatibility throughout the morph. Due to this rotation, 24 exterior Steiner vertices are used in order to achieve a natural morph.

The second example is a morph of the letter U to the letter S. The two input polygons contain 30 vertices each, and another 69 interior and 3043 exterior Steiner vertices are generated by our system (see Fig. 7). It required a few seconds on a 233 MHz PC to generate each frame of the morph.

Fig. 8 contains some examples of morphed polygons, where the triangulated interiors were exploited for texture mapping.

6. Discussion and conclusion

This paper has shown how to morph simple polygons in a manner that guarantees that the intermediate shapes are also simple. This is the first morphing method which relies on a solid mathematical (as opposed to heuristic) basis to achieve this. It is also a completely automatic process.

A number of parameters related to the morph of the compatible triangulations generated during the process provide some control over the specific morph generated. Another algorithmic parameter which might affect the quality of the morph is the size of the convex enclosure. Right now we construct the convex enclosure such that its diameter is approximately twice that of the maximum of the source and target polygon diameters. This seems to be the natural choice, as less than this would over-constrain the vertex trajectories, and more than this would not add any extra flexibility.

Many interactive morphing systems allow the user some control over the vertex trajectories, e.g. by specifying the locations of some of the vertices in some of the intermediate frames. While not demonstrated here, our method supports this since the underlying techniques of Surazhsky and Gotsman [17] for morphing compatible triangulations also support it.

A central ingredient in our method is the compatible triangulation of the source and target polygons using a small number of Steiner vertices and maintaining this compatibility throughout the morph. The fact that the triangulation is valid throughout the morph has the extra benefit that texture mapping is well defined in the polygon interior (as well as its exterior) throughout the morph, so the method may be used to morph images while avoiding the common foldover artifacts that occur in many other methods.

Our method of compatible triangulating simple polygons and polygons with one hole is of independent interest, and generates quite fair triangulations using a small number of Steiner vertices. In general there is a tradeoff between the number of Steiner points and the fairness of the triangulation, in the sense that a very small number of Steiner points will not allow a very fair triangulation, and a large number of Steiner points gives a lot of flexibility. The crux of the matter is to carefully position as few Steiner points as possible in order to optimize the tradeoff. We are able to do this by taking into account the relative shapes and feature points of the input polygons. Reducing the number of Steiner points reduces the complexity of the morphing problem, and the number of compatibility constraints. A more detailed description of the triangulation algorithms and some enhancements, which further reduce the number of Steiner vertices, along with formal correctness proofs, will be published separately [22].

In terms of complexity, most of the morphing computation time is spent on solving a set of linear equations per intermediate frame. Our implementation has not been optimized to solve these equations efficiently. However, Floater [8] has reported that these sets of equations may be solved in an order of magnitude less CPU time by using sparse matrix numerical methods.

Future work includes generalizing our algorithm to more complex planar shapes, such as polylines, and the so-called "skeletons" or "stick figures" — connected straight-line planar graphs. Morphing between unconnected planar figures, or between figures with completely different topologies, is more complicated.

The most challenging task ahead, by far, is to generalize our methods to 3D, namely, to morph 3D polyhedra. The straightforward extension to polyhedra involves compatible tetrahedralizations of the polyhedra, which seems to be much more difficult than triangulation of planar polygons. We believe, however, that the theory and basic methods should carry through.

References

[1] Beier T, Neely S. Feature-based image metamorphosis. Computer Graphics (SIGGRAPH '92) 1992;26:35–42.

[2] Fujimura K, Makarov M. Foldover-free image warping. Graphical Models and Image Processing 1998;60:100–11.

[3] Tal A, Elber G. Image morphing with feature preserving texture. Computer Graphics Forum (Eurographics '99 Proceedings) 1999;18:339–48.

[4] Alexa M, Cohen-Or D, Levin D. As-rigid-as-possible polygon morphing. Proceedings of the SIGGRAPH '2000, New Orleans, 2000. p. 157–64.

[5] Carmel E, Cohen-Or D. Warp-guided object-space morphing. The Visual Computer 1997;13:465–78.

[6] Goldstein E, Gotsman C. Polygon morphing using a multiresolution representation. Proceeding of Graphics Interface, Quebec City, 1995.

[7] Sederberg TW, Gao P, Wang G, Mu H. 2D shape blending: an intrinsic solution to the vertex path problem. Computer Graphics (SIGGRAPH '93) 1993;27:15–8.

[8] Sederberg TW, Greenwood E. A physically based approach to 2D shape blending. Computer Graphics (SIGGRAPH '92) 1992;26:25–34.

[9] Shapira M, Rappoport A. Shape blending using the star-skeleton representation. IEEE Transactions on Computer Graphics and Application 1995;15:44–51.

[10] Samoilov T, Elber G. Self-intersection elimination in metamorphosis of two-dimensional curves. The Visual Computer 1998;14:415–28.

[11] Cohen-Or D, Levin D, Solomovici A. Three-dimensional distance field metamorphosis. ACM Transactions on Graphics 1998;17:116–41.

[12] Guibas L, Hershberger J, Suri S. Morphing simple polygons. Discrete and Computational Geometry 2000;24:1–34.

[13] Saalfeld A. Joint triangulations and triangulation maps. Proceedings of Third Annual ACM Symposium on Computational Geometry 1987; p. 195–204.

[14] Floater MS, Gotsman C. How to morph tilings injectively. Computational and Applied Mathematics 1999;101:117–29.

[15] Tutte WT. How to draw a graph. Proceedings of London Mathematical Society 1963;13:743–68.

[16] Floater MS. Parameterization and smooth approximation of surface triangulation. Computer Aided Geometric Design 1997;14:231–50.

[17] Surazhsky V, Gotsman C. Morphing planar triangulations. 1999, submitted for publication.

[18] Cohen S, Elber G, Bar-Yehuda R. Matching of freeform curves. Computer Aided Design 1997;29:369–78.

[19] Aronov B, Seidel R, Souvaine DL. On compatible triangulations of simple polygons. Computational Geometry: Theory and Applications 1992;3:27–35.

[20] Babikov M, Souvaine DL, Wenger R. Constructing piecewise linear homeomorphisms of polygons with holes. Proceedings of Ninth Canadian Conference on Computational Geometry, 1997.

[21] Kranakis E, Urrutia J. Isomorphic triangulations with small number of Steiner points. International Journal of Computational Geometry & Applications 1999;9:171–80.

[22] Surazhsky V, Gotsman C. Optimized compatible triangulation of polygons. Preprint, 2000.